1.   (Currently Amended) A system comprising:

a cluster having a first instance and a second instance, each of the first and second instances including a plurality of server nodes;

a control logic to start each instance by initiating a launch logic for each of the server nodes, the launch logic, when initiated, to execute Java processes in each respective server node, the control logic to reassign a failed Java process previously running on a first server node to a second server node; and

a communication interface coupled between the launch logic and the control logic to enable the launch logic to obtain status of each of the Java processes and enable the control logic to access the status in a shared memory via the communication interface, the launch logic to store and maintain the status in the shared memory via the communication interface.

2.   (Original) The system of claim 1, wherein the launch logic is provided to load a virtual machine and execute a Java process in the virtual machine.

3.   (Previously Presented) The system of claim 2, wherein the communication interface comprises:

the shared memory to store the status of the Java processes.

4.   (Original) The system of claim 3, wherein the launch logic comprises:

a Java native interface to obtain the status of each of the Java processes and to update the shared memory with the obtained status.

5.   (Original) The system of claim 4, wherein the control logic accesses the shared memory to monitor the status of each of the Java processes.

6.   (Original) The system of claim 1, wherein the control logic is provided to detect a failure of a Java process and to automatically restart the failed Java process.

7.   (Original) The system of claim 1, wherein the control logic is provided to generate an instruction to start, terminate or restart a particular process executed server nodes based on a command received from a remote device.

8.   (Original) The system of claim 1, wherein the communication interface further comprises:

a named pipe to send and receive commands between the control logic and the launch logic.

9.   (Original) The system of claim 1, wherein the control logic comprises:

a signal handler to receive and interpret signals from a management console.

10.   (Original) The system of claim 1, wherein the control logic comprises:

a server connector to enable connection with an external server.

11.   (Original) The system of claim 1, wherein the control logic comprises:

Java native processes.

12.   (Original) The system of claim 1, wherein the launch logic comprises:

a container combining Java native processes with a Java virtual machine.

13.   (Currently Amended) A method comprising:

executing Java processes for a plurality of server nodes in an instance;

obtaining status regarding the Java processes executed by the server nodes in the instance;

storing the status regarding the Java processes in a communication interface, the communication interface updating and maintaining the status in a shared memory;

accessing the status in the shared memory via the communication interface; and

reassigning a failed Java process previously running on a first server node to a second server node.

14.   (Original) The method of claim 13, further comprising:

enabling control of the Java processes based on an instruction received from a remote device.

15.   (Original) The method of claim 13, further comprising:

using a Java native interface to obtain the status regarding the Java processes.

16.   (Original) The method of claim 13, further comprising: detecting a failure of a process within the cluster by accessing the status in the communication interface; and

restarting the failed process.

17.   (Currently Amended) A machine-readable medium that provides instructions, which when executed by a processor cause the processor to perform operations comprising:

executing Java processes for a plurality of server nodes in an instance;

obtaining status regarding each of the Java processes executed by the server nodes in the instance; ~~and~~

storing the status regarding the Java processes into a shared memory; and

reassigning a terminated Java process from a first server node to a new server node.

18.   (Original) The machine-readable medium of claim 17, wherein the operations performed by the processor further comprise:

invoking a Java native interface to obtain the status regarding the Java processes.

19.   (Original) The machine-readable medium of claim 17, wherein the operations performed by the processor further comprise:

receiving instructions via a communication interface; and

starting, terminating or restarting a process based on the instructions received via the communication interface.

20.   (Original) The machine-readable medium of claim 17, wherein the operations further comprise: detecting a failure of a process within the cluster by accessing the status in the shared memory and automatically restarting the failed process.

21.   (Currently Amended) An apparatus comprising:

a cluster having a first instance and a second instance, each of the first and second instances including a plurality of server nodes;

a control logic to start each respective instance by initiating a launch logic for each respective server node in the first and second instances;

the launch logic, for each respective server node in the first and second instances, to further launch Java processes, and obtain a status of the Java processes to store and maintain in a shared memory; and

the control logic to access the status obtained by the launch logic via the shared memory, the control logic to reassign a failed Java process from a first server node to a new server node.

22.   (Previously Presented) The apparatus of claim 21, further comprising:

the shared memory to enable exchange of information between the Java processes and the control logic.

23.   (Original) The apparatus of claim 21, wherein the launch logic loads a virtual machine and executes Java processes.

24.   (Original) The apparatus of claim 23, wherein the launch logic uses a Java native interface to obtain a status of each of the Java processes, and updates information contained in the shared memory based on the status obtained by the Java native interface.

25.   (Original) The apparatus of claim 21, wherein the control logic detects a failure of a process within the cluster; and automatically restarts operations of the failed process.

26.   (Original) The apparatus of claim 21, further comprising:

a signal handler to receive a command from a remote device and controlling one of the Java processes based on the command received from the remote device.

27.   (Original) The apparatus of claim 21, further comprising:

a named pipe to send and receive commands between the control logic and the launch logic.

28.   (Currently Amended) A system comprising:

a cluster having a first instance and a second instance, each of the first and second instances including a plurality of server nodes;

means for starting each instance by executing Java processes in each respective server node; and

means for enabling exchange of information that is stored and maintained in a shared memory between the Java processes and the means for starting each instance; and

means for reassigning a failed Java process from a first server node to a second server node.

29.   (Original) The system of claim 28, further comprising:

      means for loading a virtual machine and execute a Java process in the virtual machine.

30.   (Previously Presented) The system of claim 28, wherein the means for enabling exchange of information comprises:

      the shared memory having a plurality of entries.

31.   (Original) The system of claim 30, further comprising:

      means for obtaining status for each of the Java processes; and

      means for updating the shared memory with the obtained status.

32.   (Original) The system of claim 31, further comprising:

      means for accessing the shared memory to monitor the status of each of the Java processes; and

      means for sending an instruction to the launch means to start, terminate or restart a particular process executed in the cluster.

33.   (Original) The system of claim 28, further comprising:

      means for enabling a user to monitor and control the Java processes running in the cluster from a management console coupled to the means for controlling; and

      means for enabling a connection with an external server.

34.   (Previously Presented) The system of claim 1, wherein the launch logic stores the status independent of the control logic accessing the status.

35.   (Previously Presented) The system of claim 1, wherein a persistent data structure is stored in the shared memory to enable an independent exchange of information between the Java processes.

36.   (Previously Presented) The method of claim 13, wherein accessing the status occurs independent of storing the status.

37.   (Previously Presented) The machine-readable medium of claim 17, wherein storing the status occurs independent of accessing the status:

38.   (Previously Presented) The apparatus of claim 21, wherein the control logic accesses the status independent of the launch logic storing the status.

39.   (Previously Presented) The system of claim 28, wherein the exchange of information is achieved by an independent storing and an independent accessing of the information.